# Runtime Monitoring with R2U2 for Aircraft Systems with Neural Networks

Johann Schumann[†]

[†]KBR/Wyle, NASA Ames Research Center

Johann.M.Schumann@nasa.gov

# Runtime Monitoring

$$R_1: \quad \text{BeginPresentation} \longrightarrow \Diamond_{[45min,48min]} \text{"The End"}$$

# Runtime Monitoring

$$R_1: \quad \text{BeginPresentation} \longrightarrow \Diamond_{[45min,48min]} \text{"The End"}$$

R2U2

- **R**ESPONSIVENESS: respond in "real time"
- **R**EALIZABILITY: plug-and-play
- **U**NOBTRUSIVENESS: do not "mess up" the flight software
- **U**NIT

R2U2 is a run-time monitoring and V&V tool that combines *Metric Temporal Logic* observers, *Bayesian Network* reasoners, and *model-based prognostics*.

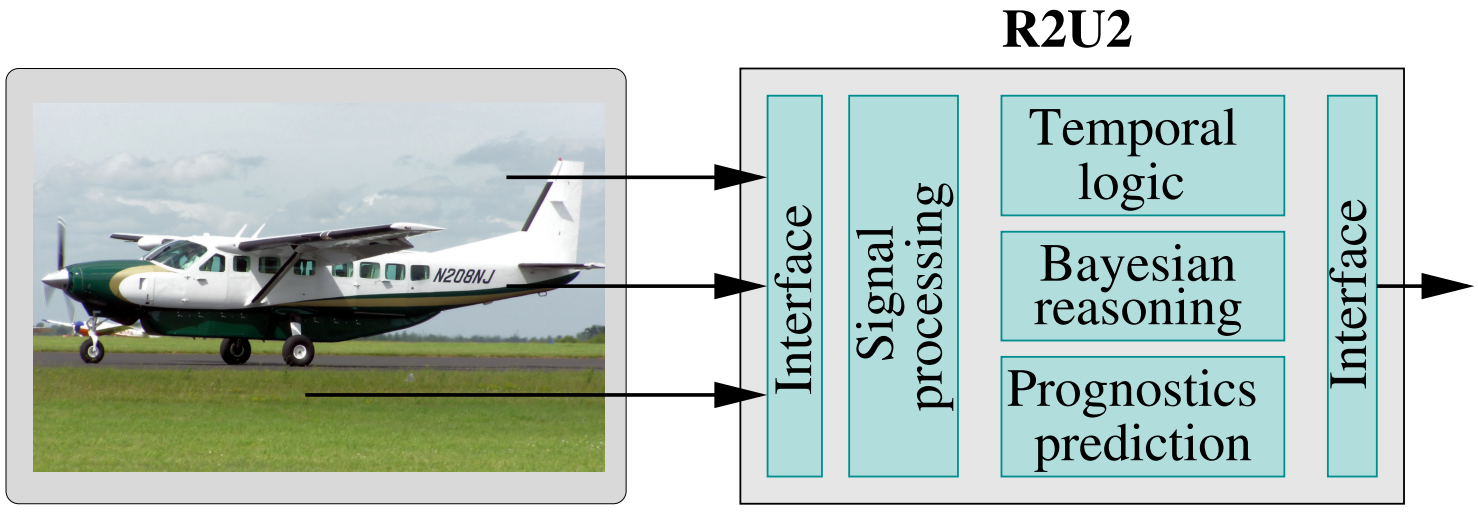
# Introduction

On a flight from Boston, Mass. to SFO...


Software Error? Sensor Failure? Operational Error?

Pilot: "The sun is on the left hand side, so we are OK"

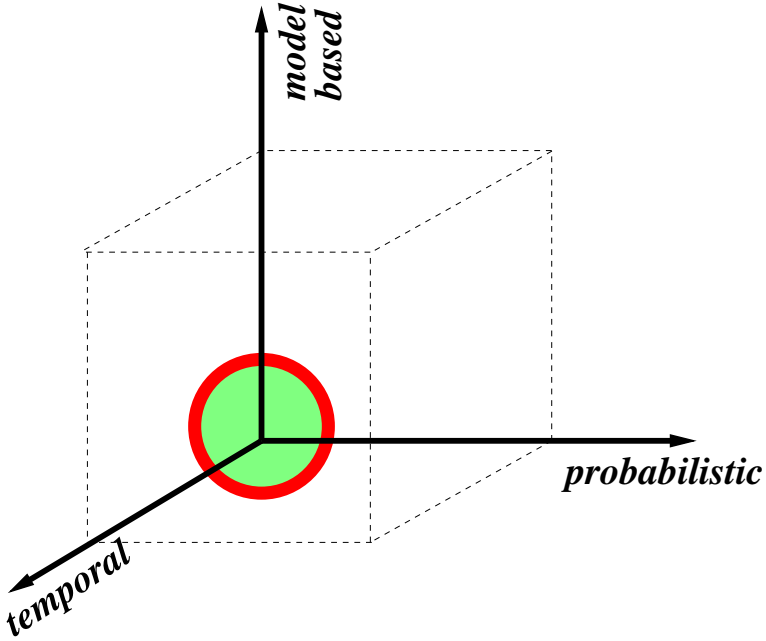
# The R2U2 Runtime Monitoring System




R2U2 is a run-time monitoring and V&V tool that combines *Metric Temporal Logic* observers, *Bayesian Network* reasoners, and *model-based prognostics*.

# Why all the R2U2 Ingredients?

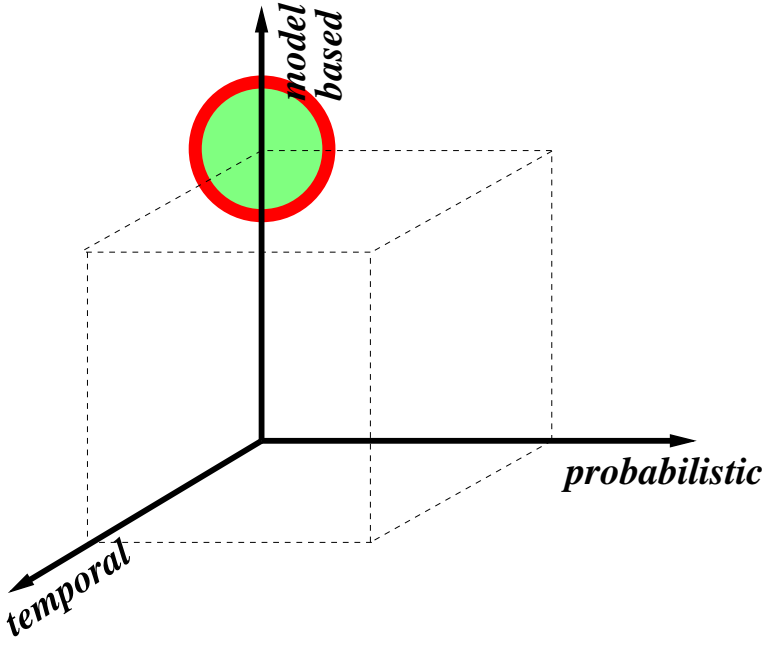
Different health management approaches focus on specific properties




- Boolean Logic (assertions)
  - value and rate checkers
  - thresholding

# Why all the R2U2 Ingredients?

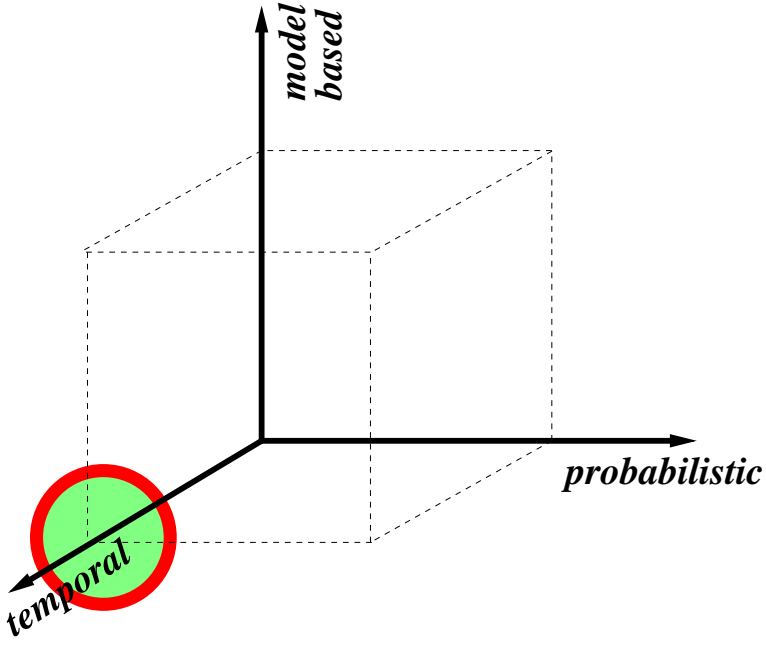
Different health management approaches focus on specific properties




- Boolean Logic (assertions)
  - value and rate checkers
  - thresholding
- Model-Based Diagnostics and Monitoring
  - TEAMS
  - Prognostics

# Why all the R2U2 Ingredients?

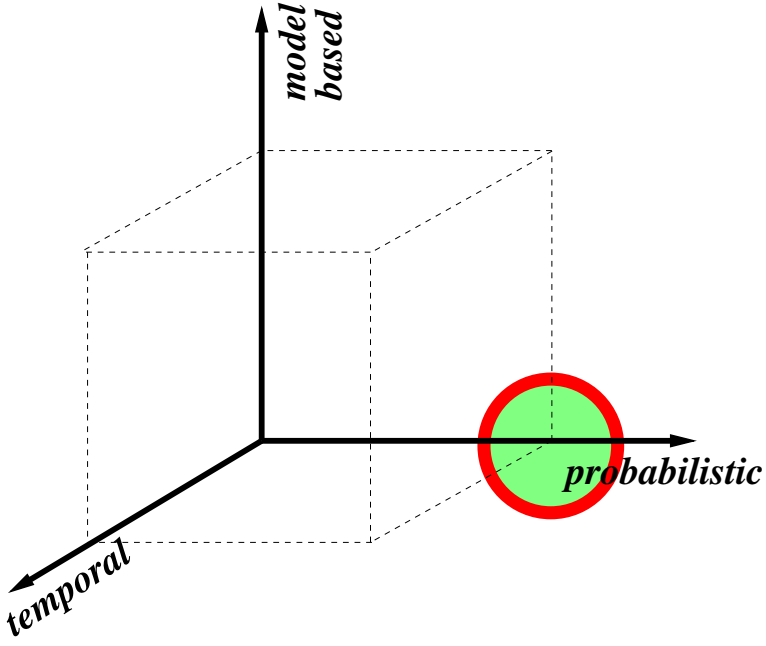
Different health management approaches focus on specific properties




- Boolean Logic (assertions)
  - value and rate checkers
  - thresholding
- Model-Based Diagnostics and Monitoring
  - TEAMS
  - Prognostics
- Temporal
  - TFPG, checkers with delays

# Why all the R2U2 Ingredients?

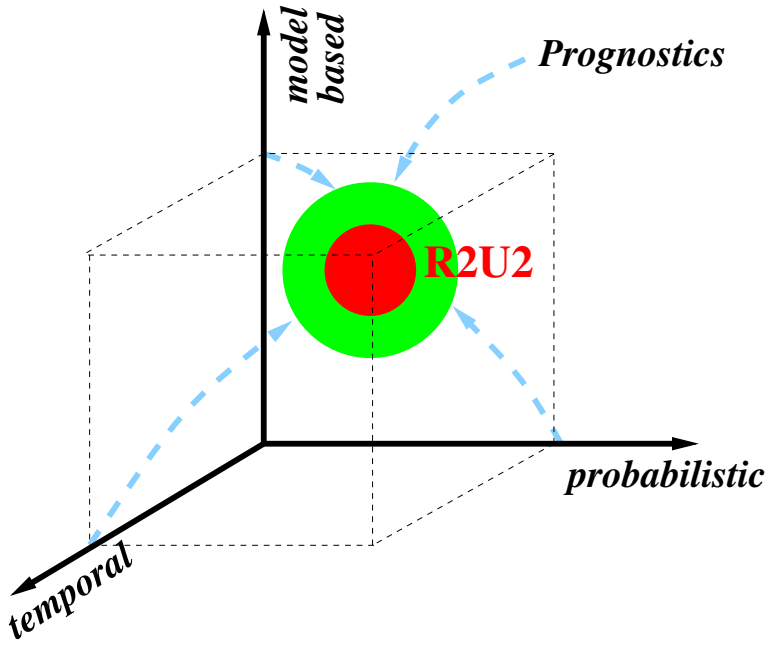
Different health management approaches focus on specific properties




- Boolean Logic (assertions)
  - value and rate checkers
  - thresholding
- Model-Based Diagnostics and Monitoring
  - TEAMS
  - Prognostics
- Temporal
  - TFPG, checkers with delays
- Probabilistic
  - Bayesian Networks

# Why all the R2U2 Ingredients?

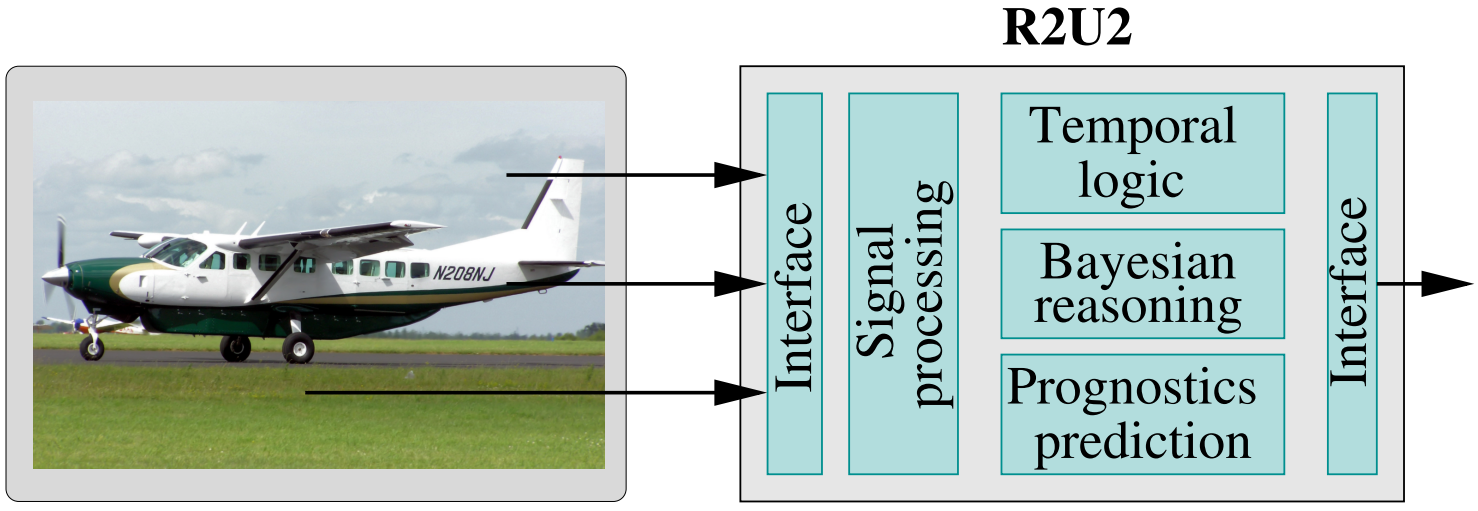
Different health management approaches focus on specific properties




- Boolean Logic (assertions)
  - value and rate checkers
  - thresholding
- Model-Based Diagnostics and Monitoring
  - TEAMS
  - Prognostics
- Temporal
  - TFPG, checkers with delays
- Probabilistic
  - Bayesian Networks

Our R2U2 framework provides powerful mechanisms to enable temporal, probabilistic diagnostic models integrating advanced prognostics models.

# R2U2 Architecture




- R2U2 as software node for ROS
- R2U2 as software app for NASA cFS/cFE Core Flight System
- R2U2 as Simulink Block
- R2U2 as Field Programmable Gate Array (FPGA) configuration

# Ingredient I: Metric Temporal Logic

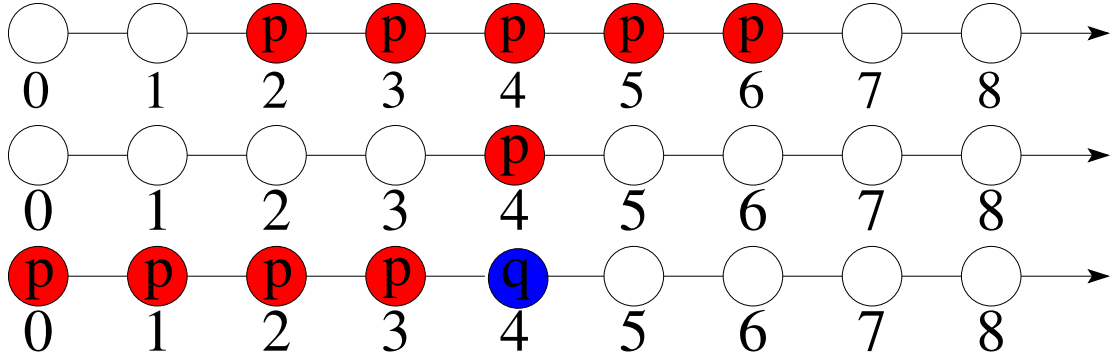
**Future Time Metric Temporal Logic** (MTL) reasons about *bounded* timelines:

- Atomic propositions: $p, q$
- Operators: $\neg$, $\wedge$, $\vee$, $\rightarrow$, $\leftrightarrow$, $\circ$,$\Box_{\mathcal{I}}$, $\Diamond_{\mathcal{I}}$, $\mathcal{U}_{\mathcal{I}}$, $\mathcal{R}_{\mathcal{I}}$

$\Box_{[2,6]} p$ ALWAYS$_{[2,6]}$

$\Diamond_{[2,6]} p$ EVENTUALLY$_{[0,7]}$

$p \mathcal{U} q$ UNTIL




- **Past Time**: similar with temporal operators Historically, Once, Since

# Observer Pairs

For each future time MTL formula, we create two observers:

- *asynchronous* observers return $\{T, F\}$
  - results are not instantaneous
  - observer has considerable complexity and needs local memory
- *synchronous* observer returns $\{T, F, \textbf{maybe}\}$ at each timestamp
  - results immediately available
  - observer has low complexity
  - three-valued logic can be useful for reasoning

We do not translate MTL formulas into finite state machines; rather we use synchronisation queues [TACAS'2014]

# Example: Safety Rule

After receiving a command for takeoff, the UAS shall reach an altitude of 600ft within 60 seconds.

# Example: Safety Rule

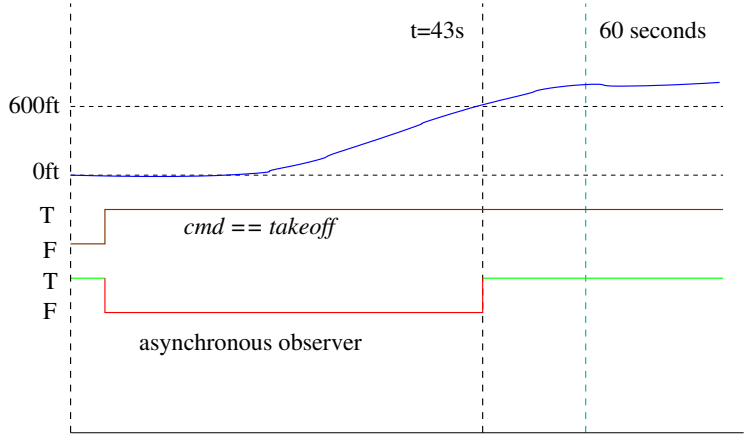
After receiving a command for takeoff, the UAS shall reach an altitude of 600ft within 60 seconds.

$$\square((\text{cmd} == \text{takeoff}) \rightarrow \Diamond_{[0,60s]}(alt \geq 600 \text{ ft}))$$

# Example: Safety Rule

After receiving a command for takeoff, the UAS shall reach an altitude of 600ft within 60 seconds.

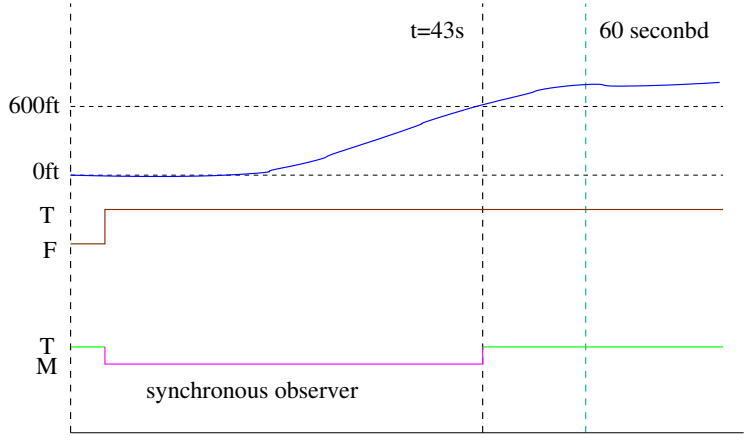
$$\square((\text{cmd} == \text{takeoff}) \rightarrow \Diamond_{[0,60s]}(alt \geq 600 \text{ ft}))$$



# Example: Safety Rule

After receiving a command for takeoff, the UAS shall reach an altitude of 600ft within 60 seconds.

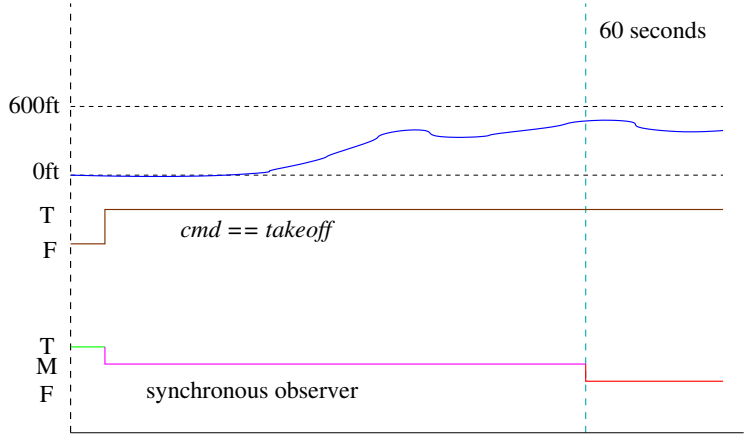
$$\square((\text{cmd} == \text{takeoff}) \rightarrow \Diamond_{[0,60s]}(alt \geq 600 \text{ ft}))$$



# Example: Safety Rule

After receiving a command for takeoff, the UAS shall reach an altitude of 600ft within 60 seconds.

$$\square((\text{cmd} == \text{takeoff}) \rightarrow \Diamond_{[0,60s]}(alt \geq 600 \text{ ft}))$$



# Ingredient III: Model-based Prognostics

Is there enough battery to fly over the hill? Flight time is 10 minutes

$$\text{start-climb} \longrightarrow \Box_{[10min]}\text{battery-OK}$$

# Ingredient III: Model-based Prognostics

Is there enough battery to fly over the hill? Flight time is 10 minutes

$$\text{start-climb} \longrightarrow \Box_{[10min]}\text{battery-OK}$$

- Not Good: valuation only availabe after 10 minutes

# Ingredient III: Model-based Prognostics

Is there enough battery to fly over the hill? Flight time is 10 minutes

$$\text{start-climb} \longrightarrow \Box_{[10min]}\text{battery-OK}$$

- Not Good: valuation only availabe after 10 minutes
- Not much better: MAYBE Valuation available now

# Ingredient III: Model-based Prognostics

Is there enough battery to fly over the hill? Flight time is 10 minutes

$$\text{start-climb} \longrightarrow \Box_{[10min]}\text{battery-OK}$$

- Not Good: valuation only availabe after 10 minutes
- Not much better: MAYBE Valuation available now

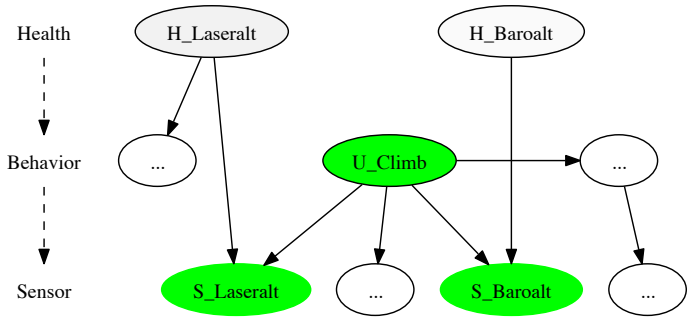
$$\text{start-climb} \longrightarrow (RUL > 10min)$$

- Good: model-based prognostics. Result available *now*

R2U2 uses electro-chemical model for LiPo batteries and a UKF-based algorithm

# Bayesian Reasoning

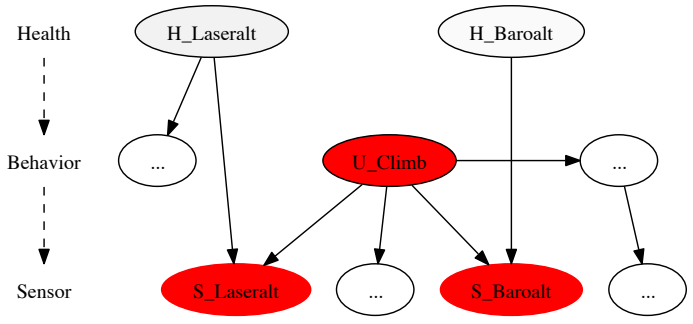
- Bayesian models for diagnostic reasoning and health management are well established
- Our Bayesian Networks (BNs) contain
  - health nodes $H$ ("output")
  - behavior nodes
  - observable sensor nodes $S$




R2U2 provides efficient constant-footprint implementation and tight integration with temporal properties

# Bayesian Reasoning

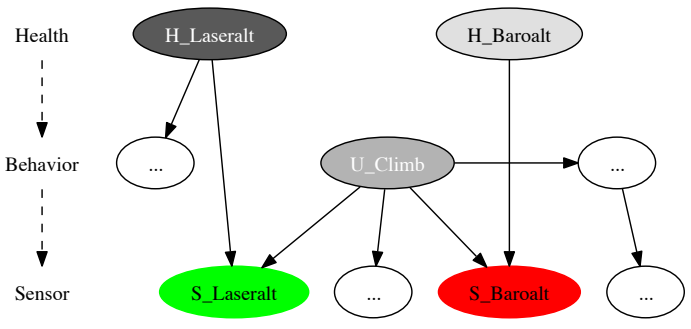
- Bayesian models for diagnostic reasoning and health management are well established
- Our Bayesian Networks (BNs) contain
  - health nodes $H$ ("output")
  - behavior nodes
  - observable sensor nodes $S$




R2U2 provides efficient constant-footprint implementation and tight integration with temporal properties

# Bayesian Reasoning

- Bayesian models for diagnostic reasoning and health management are well established
- Our Bayesian Networks (BNs) contain
  - health nodes $H$ ("output")
  - behavior nodes
  - observable sensor nodes $S$




R2U2 provides efficient constant-footprint implementation and tight integration with temporal properties

# R2U2 Capabilities and Applications

- Signal Processing
- Past Time Temporal Logic
- Future Time Temporal Logic
- Bayesian Reasoning
- Prognostics

# R2U2 Capabilities and Applications

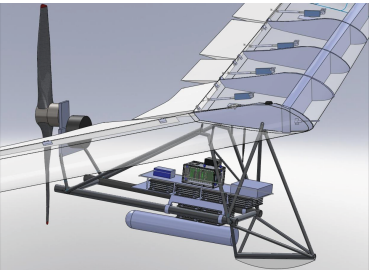
- Signal Processing
- Past Time Temporal Logic
- Future Time Temporal Logic
- Bayesian Reasoning
- Prognostics

- safety monitoring
- performance monitoring
- security monitoring
- failure diagnosis
- prognostics
- autonomous decision making

# R2U2 Capabilities and Applications

- Signal Processing
- Past Time Temporal Logic
- Future Time Temporal Logic
- Bayesian Reasoning
- Prognostics

- safety monitoring
- performance monitoring
- security monitoring
- failure diagnosis
- prognostics
- autonomous decision making



Swift, ARC

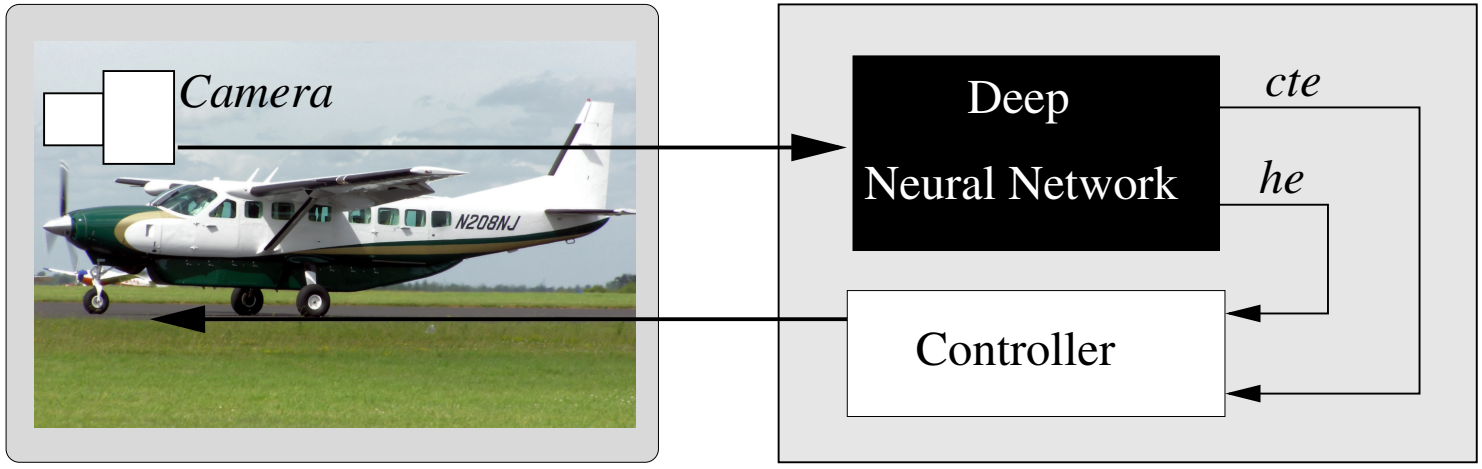
Dragoneye, ARC


Edge 540, LaRC

AOS, ARC

# R2U2 for Monitoring of Neural Networks

Case Study: Autonomous Center Line Tracking




- NN can produce noisy or arbitrary result
- NN can produce wrong result
- NN is inherently probabilistic
- NN lacks explainability and interpretability

runtime monitoring on NN level and system level is necessary

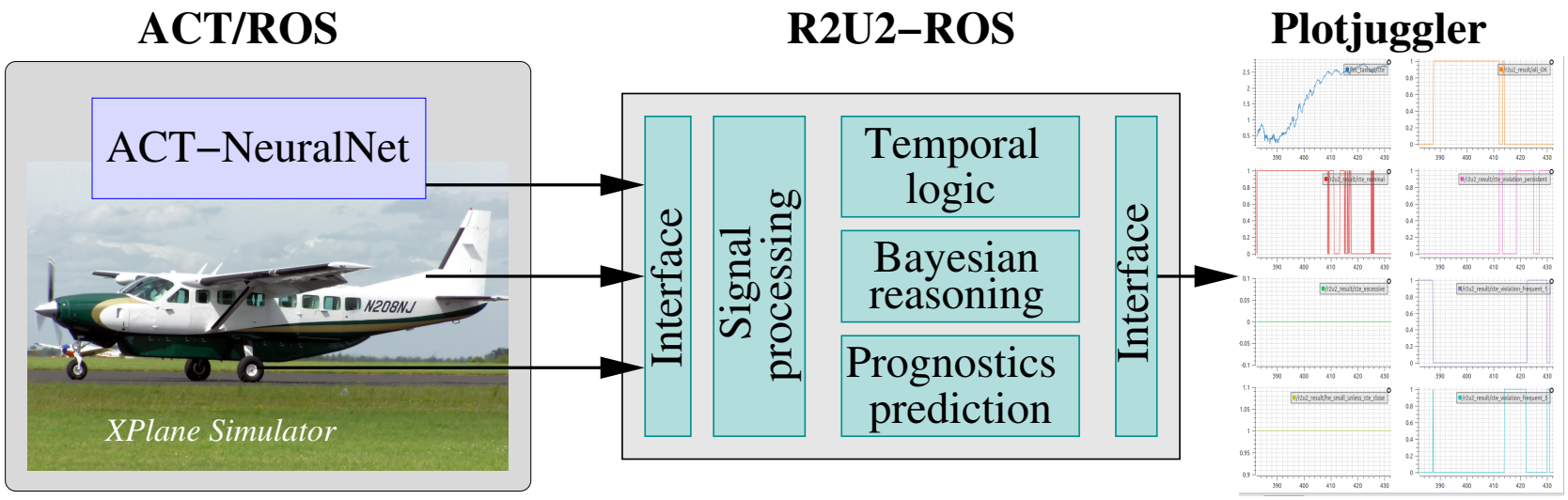
# R2U2 support

R2U2 can

- analyze and incorporate signals from NN-specific sensors (e.g., from Prophecy runtime)
- use results of Bayesian NN calculations for safety reasoning (e.g., confidence of outputs or weights)
- detect and diagnose NN related failures (e.g., camera sensor problems) using model-based diagnostics
- merge system sensor signals with NN signals (sensor fusion)
- monitor temporal behavior on system level and on component level
- monitor behavior for onboard learning applications (N/A here)

# R2U2 on ACT

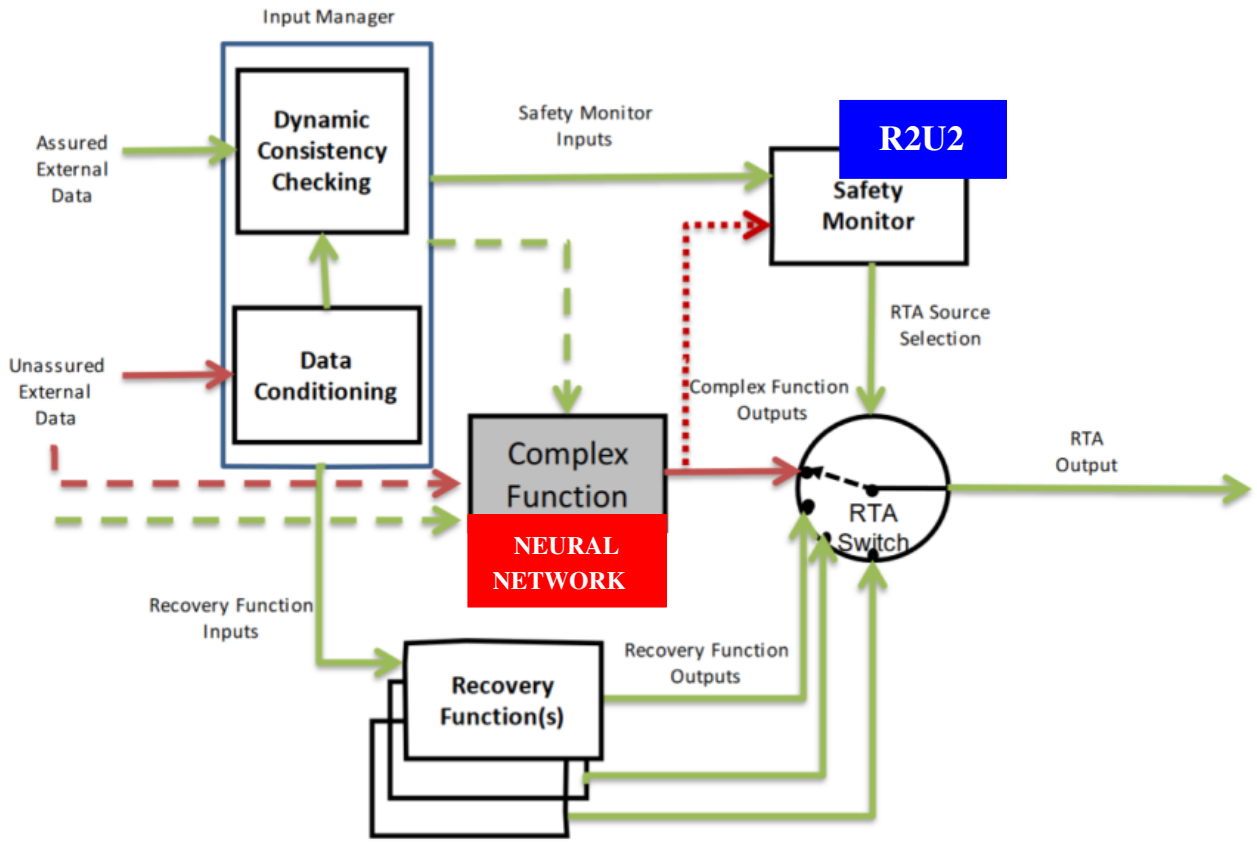
R2U2 as a ROS node monitoring the behavior of ACT




video/video_1.mp4

# Toward a Runtime Assurance Architecture (Future Work)

R2U2 can be Safety Monitor in an ASTM F3269 style system




underlying figure from P. Nagarajan, S. Kannan, et.al. *SciTech 2021*

# Summary

- R2U2 is a Responsive, Realizable, and Unobtrusive Unit for system and component runtime monitoring
- R2U2 combines *Metric Temporal Logic* observers, *Bayesian* reasoners, and *model-based prognostics*
- R2U2 implementations for ROS, Simulink, cFS, and FPGA
- R2U2 applicable for safety monitoring, performance monitoring, security monitoring, failure diagnosis, prognostics, and autonomous decision making
- R2U2 useful for monitoring of systems with Neural Networks

# Selected Publications


J. Schumann, et.al. R2U2 Manual, 2018.

K. Y. Rozier and J. Schumann. R2U2: Tool Overview RV-CuBES 2017.

P. Moosbrugger, K. Y. Rozier, and J. Schumann. R2U2: Monitoring and Diagnosis of Security Threats for Unmanned Aerial Systems. Formal Methods in System Design, 2017.

J. Schumann, I. Roychoudhury, and C. Kulkarni: Diagnostic Reasoning using Prognostic Information for Unmanned Aerial Systems. In PHM-2015, 2015.

J. Schumann, K. Y. Rozier, T. Reinbacher, O. J. Mengshoel, T. Mbaya, and C. Ippolito: Towards Real-time, On-board, Hardware-supported Sensor and Software Health Management for Unmanned Aerial Systems. IJPHM, 2015.

J. Geist, K. Y. Rozier and J. Schumann: Runtime Observer Pairs and Bayesian Network Reasoners On-board FPGAs: Flight-Certifiable System Health Management for Embedded Systems In RV 2014, Springer, 2014.

T. Reinbacher and K. Y. Rozier and J. Schumann: Hardware Enabled Unobtrusive Health Monitoring of Real-Time Systems. In TACAS 2014, Springer, 2014.

A. Srivastava and J. Schumann: Software Health Management: A Necessity for Safety Critical Systems. Innovations in Systems and Software Engineering, Springer, 9(4):219–233, 2013.

# Team






- Johannes Geist
- Timmy Mbaya
- Thomas Reinbacher
- Julian Rhein
- Kristin Rozier
- Johann Schumann

PoC: `johann.m.schumann@nasa.gov` or `johann.schumann@us.kbr.com`

Thank You!